15-462 Computer Graphics I

Lecture 9

Curves and Surfaces

Parametric Representations
Cubic Polynomial Forms
Hermite Curves
Bezier Curves and Surfaces
[Angel 10.1-10.6]

February 11, 2003
Frank Pfenning
Carnegie Mellon University

http://www.cs.cmu.edu/~fp/courses/graphics/

Goals

- How do we draw surfaces?
  - Approximate with polygons
  - Draw polygons
- How do we specify a surface?
  - Explicit, implicit, parametric
- How do we approximate a surface?
  - Interpolation (use only points)
  - Hermite (use points and tangents)
  - Bezier (use points, and more points for tangents)
- Next lecture: splines, realization in OpenGL

Explicit Representation

- Curve in 2D: $y = f(x)$
- Curve in 3D: $y = f(x)$, $z = g(x)$
- Surface in 3D: $z = f(x,y)$
- Problems:
  - How about a vertical line $x = c$ as $y = f(x)$?
  - Circle $y = \pm (r^2 - x^2)^{1/2}$ two or zero values for x
- Too dependent on coordinate system
- Rarely used in computer graphics

Implicit Representation

- Curve in 2D: $f(x,y) = 0$
  - Line: $ax + by + c = 0$
  - Circle: $x^2 + y^2 - r^2 = 0$
- Surface in 3d: $f(x,y,z) = 0$
  - Plane: $ax + by + cz + d = 0$
  - Sphere: $x^2 + y^2 + z^2 - r^2 = 0$
- $f(x,y,z)$ can describe 3D object:
  - Inside: $f(x,y,z) < 0$
  - Surface: $f(x,y,z) = 0$
  - Outside: $f(x,y,z) > 0$

Algebraic Surfaces

- Special case of implicit representation
- f(x,y,z) is polynomial in x, y, z
- Quadrics: degree of polynomial $\leq$ 2
- Render more efficiently than arbitrary surfaces
- Implicit form often used in computer graphics
- How do we represent curves implicitly?

Parametric Form for Curves

- Curves: single parameter u (e.g. time)
- x = x(u), y = y(u), z = z(u)
- Circle: x = cos(u), y = sin(u), z = 0
- Tangent described by derivative

$$\mathbf{p}(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} \qquad \frac{d\mathbf{p}(u)}{du} = \begin{bmatrix} \frac{dx(u)}{du} \\ \frac{dy(u)}{du} \\ \frac{dz(u)}{du} \end{bmatrix}$$

- Magnitude is “velocity”

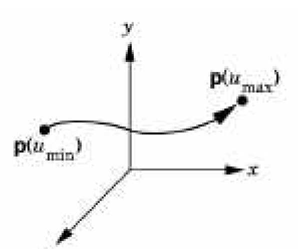
Parametric Form for Surfaces

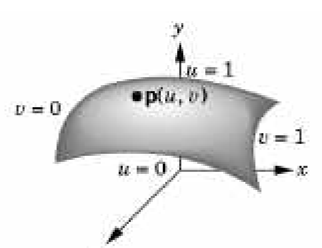
- Use parameters u and v
- x = x(u,v), y = y(u,v), z = z(u,v)
- Describes surface as both u and v vary
- Partial derivatives describe tangent plane at each point p(u,v) = [x(u,v) y(u,v) z(u,v)]^T

$$\frac{\partial \mathbf{p}(u,v)}{\partial u} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial u} \\ \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial z(u,v)}{\partial u} \end{bmatrix} \qquad \frac{\partial \mathbf{p}(u,v)}{\partial v} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial u} \\ \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial z(u,v)}{\partial u} \end{bmatrix}$$

Assessment of Parametric Forms

- Parameters often have natural meaning
- Easy to define and calculate
  - Tangent and normal
  - Curves segments (for example, $0 \leq u \leq 1$)
  - Surface patches (for example, $0 \leq u,v \leq 1$)





Parametric Polynomial Curves

- Restrict x(u), y(u), z(u) to be polynomial in u
- Fix degree n

$$\mathbf{p}(u) = \sum_{k=0}^{n} \mathbf{c}_k u^k$$

- Each c_k is a column vector

$$\mathbf{c}_k = \begin{bmatrix} c_{xk} \\ c_{yk} \\ c_{zk} \end{bmatrix}$$

Parametric Polynomial Surfaces

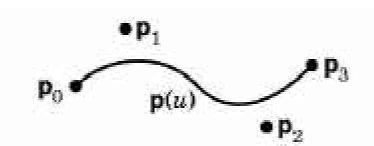
- Restrict x(u,v), y(u,v), z(u,v) to be polynomial of fixed degree n

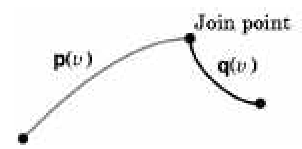
$$\mathbf{p}(u,v) = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix} = \sum_{i=0}^{n} \sum_{k=0}^{n} \mathbf{c}_{ik} u^i v^k$$

- Each c_{ik} is a 3-element column vector
- Restrict to simple case where $0 \leq u,v \leq 1$

Approximating Surfaces

- Use parametric polynomial surfaces
- Important concepts:
  - Join points for segments and patches
  - Control points to interpolate
  - Tangents and smoothness
  - Blending functions to describe interpolation
- First curves, then surfaces





Outline

- Parametric Representations
- Cubic Polynomial Forms
- Hermite Curves
- Bezier Curves and Surfaces

Cubic Polynomial Form

- Degree 3 appears to be a useful compromise
- Curves:

$$\mathbf{p}(u) = \mathbf{c}_0 + \mathbf{c}_1 u + \mathbf{c}_2 u^2 + \mathbf{c}_3 u^3 = \sum_{k=0}^{3} \mathbf{c}_k u^k$$

- Each c_k is a column vector $[c_{kx}\ \ c_{ky}\ \ c_{kz}]^T$
- From control information (points, tangents) derive 12 values c_{kx}, c_{ky}, c_{kz} for $0 \leq k \leq 3$
- These determine cubic polynomial form
- Later: how to render

Interpolation by Cubic Polynomials

- Simplest case, although rarely used
- Curves:
  - Given 4 control points p_0, p_1, p_2, p_3
  - All should lie on curve: 12 conditions, 12 unknowns
- Space $0 \leq u \leq 1$ evenly

  $p_0 = p(0)$, $p_1 = p(1/3)$, $p_2 = p(2/3)$, $p_3 = p(1)$

Equations to Determine c_k

- Plug in values for u = 0, 1/3, 2/3, 1

$$\mathbf{p}_0 = \mathbf{p}(0) = \mathbf{c}_0$$
$$\mathbf{p}_1 = \mathbf{p}(\tfrac{1}{3}) = \mathbf{c}_0 + \tfrac{1}{3}\mathbf{c}_1 + (\tfrac{1}{3})^2\mathbf{c}_2 + (\tfrac{1}{3})^3\mathbf{c}_3$$
$$\mathbf{p}_2 = \mathbf{p}(\tfrac{2}{3}) = \mathbf{c}_0 + \tfrac{2}{3}\mathbf{c}_1 + (\tfrac{2}{3})^2\mathbf{c}_2 + (\tfrac{2}{3})^3\mathbf{c}_3$$
$$\mathbf{p}_3 = \mathbf{p}(1) = \mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{c}_3$$

$$\begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & (\frac{1}{3})^2 & (\frac{1}{3})^3 \\ 1 & \frac{2}{3} & (\frac{2}{3})^2 & (\frac{2}{3})^3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \end{bmatrix}$$

Note:
p_k and c_k are vectors!

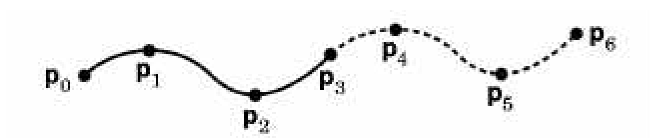
Interpolating Geometry Matrix

- Invert A to obtain interpolating geometry matrix

$$\mathbf{A}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -5.5 & 9 & -4.5 & 1 \\ 9 & -22.5 & 18 & 4.5 \\ -4.5 & 13.5 & -13.5 & 4.5 \end{bmatrix} \qquad \mathbf{c} = A^{-1}\mathbf{p}$$

Joining Interpolating Segments

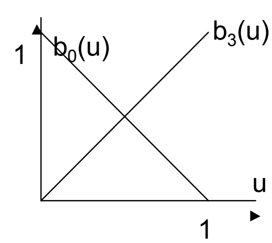
- Do not solve degree n for n points
- Divide into overlap sequences of 4 points
- p_0, p_1, p_2, p_3 then p_3, p_4, p_5, p_6, etc.



- At join points
  - Will be continuous (C^0 continuity)
  - Derivatives will usually not match (no C^1 continuity)

Blending Functions

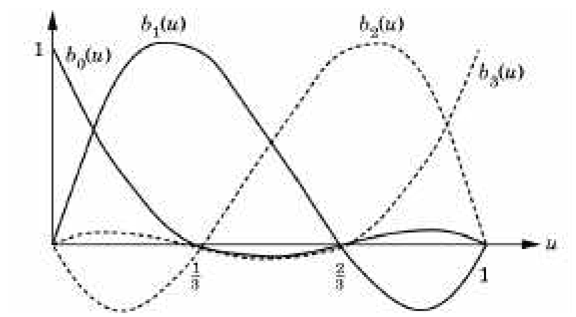
- Make explicit, how control points contribute
- Simplest example: straight line with control points p_0 and p_3
- $p(u) = (1 – u)\, p_0 + u\, p_3$
- $b_0(u) = 1 – u$, $b_3(u) = u$



Blending Polynomials for Interpolation

- Each blending polynomial is a cubic
- Solve (see [Angel, p. 427]):

$$\mathbf{p}(u) = b_0(u)\mathbf{p}_0 + b_1(u)\mathbf{p}_1 + b_2(u)\mathbf{p}_2 + b_3(u)\mathbf{p}_3$$

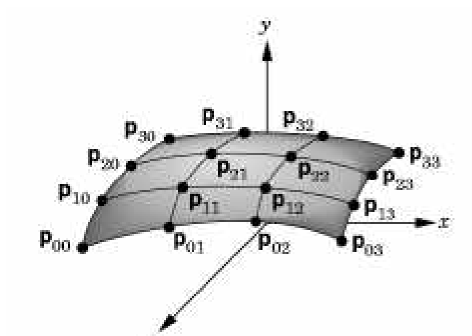


Cubic Interpolation Patch

- Bicubic surface patch with 4 × 4 control points

$$\mathbf{p}(u,v) = \sum_{i=0}^{3}\sum_{k=0}^{3} u^i v^k \mathbf{c}_{ik}$$

Note: each c_{ik} is 3 column vector (48 unknowns)



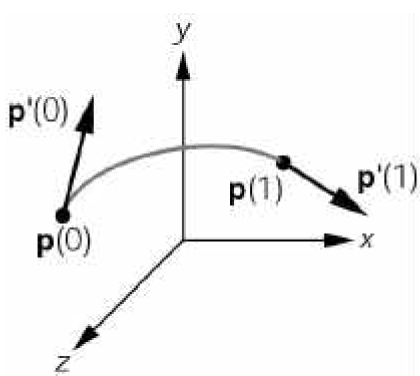
[Angel, Ch. 10.4.2]

Outline

- Parametric Representations
- Cubic Polynomial Forms
- Hermite Curves
- Bezier Curves and Surfaces

Hermite Curves

- Another cubic polynomial curve
- Specify two endpoints and their tangents



Deriving the Hermite Form

- As before

$$\mathbf{p}(0) = \mathbf{p}_0 = \mathbf{c}_0$$
$$\mathbf{p}(1) = \mathbf{p}_3 = \mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{c}_3$$

- Calculate derivative

$$\mathbf{p}'(u) = \begin{bmatrix} \frac{dx}{du} \\ \frac{dy}{du} \\ \frac{dz}{du} \end{bmatrix} = \mathbf{c}_1 + 2u\mathbf{c}_2 + 3u^2\mathbf{c}_3$$

- Yields

$$\mathbf{p}'_0 = \mathbf{p}'(0) = \mathbf{c}_1$$
$$\mathbf{p}'_3 = \mathbf{p}'(1) = \mathbf{c}_1 + 2\mathbf{c}_2 + 3\mathbf{c}_3$$

Summary of Hermite Equations

- Write in matrix form
- Remember p_k and p'_k and c_k are vectors!

$$\begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_3 \\ \mathbf{p}'_0 \\ \mathbf{p}'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \end{bmatrix}$$

- Let $q = [p_0 \;\; p_3 \;\; p'_0 \;\; p'_3]^T$ and invert to find Hermite geometry matrix M_H satisfying

$$\mathbf{c} = \mathbf{M}_H\mathbf{q}$$

Blending Functions

- Explicit Hermite geometry matrix

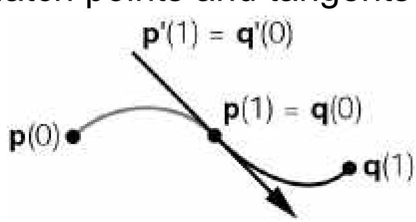
$$\mathbf{M}_H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}$$

- Blending functions for u = [1 u u^2 u^3]^T

$$\mathbf{b}(u) = \mathbf{M}_H^T \mathbf{u} = \begin{bmatrix} 2u^3 - 3u^2 + 1 \\ -2u^3 + 3u^2 \\ u^3 - 2u^2 + u \\ u^3 - u^2 \end{bmatrix}$$

Join Points for Hermite Curves

- Match points and tangents (derivates)



- Much smoother than point interpolation
- How to obtain the tangents?
- Skip Hermite surface patch
- More widely used: Bezier curves and surfaces

Parametric Continuity

- Matching endpoints (C^0 parametric continuity)

$$\mathbf{p}(1) = \begin{bmatrix} p_x(1) \\ p_y(1) \\ p_z(1) \end{bmatrix} = \begin{bmatrix} q_x(0) \\ q_y(0) \\ q_z(0) \end{bmatrix} = \mathbf{q}(0)$$

- Matching derivatives (C^1 parametric continuity)

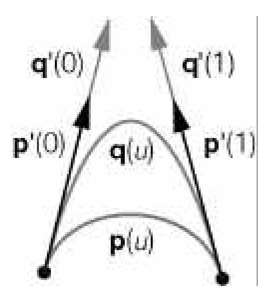
$$\mathbf{p}'(1) = \begin{bmatrix} p'_x(1) \\ p'_y(1) \\ p'_z(1) \end{bmatrix} = \begin{bmatrix} q'_x(0) \\ q'_y(0) \\ q'_z(0) \end{bmatrix} = \mathbf{q}'(0)$$

Geometric Continuity

- For matching tangents, less is required

$$\mathbf{p}'(1) = \begin{bmatrix} p'_x(1) \\ p'_y(1) \\ p'_z(1) \end{bmatrix} = k \begin{bmatrix} q'_x(0) \\ q'_y(0) \\ q'_z(0) \end{bmatrix} = k\mathbf{q}'(0)$$

- G^1 geometric continuity
- Extends to higher derivatives



Outline

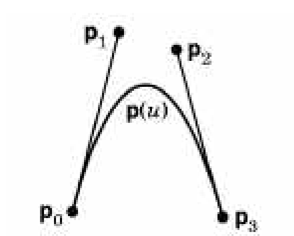
- Parametric Representations
- Cubic Polynomial Forms
- Hermite Curves
- Bezier Curves and Surfaces

Bezier Curves

- Widely used in computer graphics
- Approximate tangents by using control points

$$\mathbf{p}'(0) = 3(\mathbf{p}_1 - \mathbf{p}_0)$$

$$\mathbf{p}'(1) = 3(\mathbf{p}_3 - \mathbf{p}_2)$$



Equations for Bezier Curves

- Set up equations for cubic parametric curve
- Recall:

$$\mathbf{p}(u) = \mathbf{c}_0 + \mathbf{c}_1 u + \mathbf{c}_2 u^2 + \mathbf{c}_3 u^3$$
$$\mathbf{p}'(u) = \mathbf{c}_1 + 2\mathbf{c}_2 u + 3\mathbf{c}_3 u^2$$

- Solve for c_k

$$\begin{aligned}
\mathbf{p}_0 &= \mathbf{p}(0) = \mathbf{c}_0 \\
\mathbf{p}_3 &= \mathbf{p}(1) = \mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{c}_3 \\
\mathbf{p}'(0) &= 3\mathbf{p}_1 - 3\mathbf{p}_0 = \mathbf{c}_1 \\
\mathbf{p}'(1) &= 3\mathbf{p}_3 - 3\mathbf{p}_2 = \mathbf{c}_1 + 2\mathbf{c}_2 + 3\mathbf{c}_3
\end{aligned}$$

Bezier Geometry Matrix

- Calculate Bezier geometry matrix M_B

$$\begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \end{bmatrix} = \mathbf{M}_B \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} \text{ so } \mathbf{M}_B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

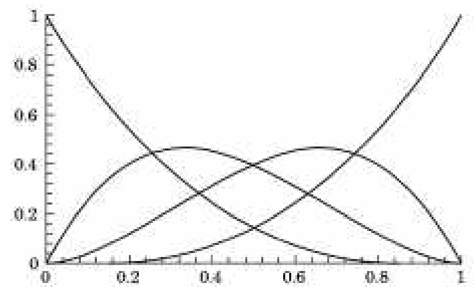
- Have C^0 continuity, not C^1 continuity
- Have C^1 continuity with additional condition

Blending Polynomials

- Determine contribution of each control point

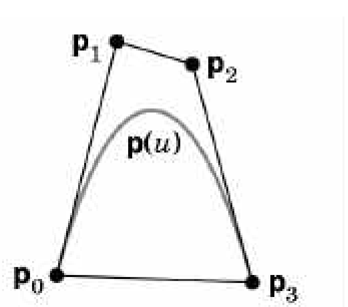
$$\mathbf{b}(u) = \mathbf{M}_B^T \mathbf{u} = \begin{bmatrix} (1-u)^3 \\ 3u(1-u)^2 \\ 3u^2(1-u) \\ u^3 \end{bmatrix}$$

Smooth blending polynomials



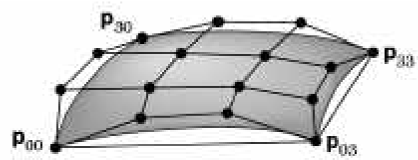
Convex Hull Property

- Bezier curve contained entirely in convex hull of control points
- Determined choice of tangent coefficient (?)



Bezier Surface Patches

- Specify Bezier patch with 4×4 control points



- Bezier curves along the boundary

$$\mathbf{p}(0,0) = \mathbf{p}_{00}$$

$$\frac{\partial \mathbf{p}}{\partial u}(0,0) = 3(\mathbf{p}_{10} - \mathbf{p}_{00})$$

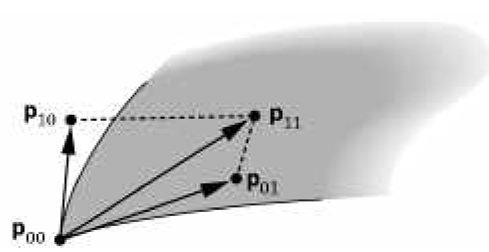
$$\frac{\partial \mathbf{p}}{\partial v}(0,0) = 3(\mathbf{p}_{01} - \mathbf{p}_{00})$$

Twist

- Inner points determine twist at corner

$$\frac{\partial^2 \mathbf{p}}{\partial u\, \partial v}(0,0) = 9(\mathbf{p}_{00} - \mathbf{p}_{01} + p_{10} - p_{11})$$

- Flat means p_{00}, p_{10}, p_{01}, p_{11} in one plane
- $(\partial^2 p/\partial u \partial v)(0,0) = 0$



Summary

- Parametric Representations
- Cubic Polynomial Forms
- Hermite Curves
- Bezier Curves and Surfaces

Preview

- B-Splines: more continuity (C^2)
- Non-uniform B-splines (“heavier” points)
- Non-uniform rational B-splines (NURBS)
  - Rational functions instead of polynomials
  - Based on homogeneous coordinates
- Rendering and recursive subdivision
- Curves and surfaces in OpenGL

Announcements

- Handing back Assignment 2 Thursday
- Model solution coming soon
- Assignment 3 due a week from Thursday
- Movie from Assignment 1!
- Thursday: Texture Mapping [Ian Graham]
- Next Tuesday: Splines